

AN OPTIMIZED HYBRID FRAMEWORK FOR REAL-TIME EDGE DETECTION AND FEATURE EXTRACTION IN AUTONOMOUS VEHICLE VISION SYSTEMS

Vivekanandan Govindan Ekambaram¹, Yamuna Natarajan²

¹Senior Data Scientist, Kforce Inc, Milpitas, California, USA

²Assistant Professor, Department of computer science, Dr. MGR Janaki college of Arts and science for women, Chennai, India

Received: 21/08/2026

Revised: 23/09/2026

Accepted: 05/10/2026

ABSTRACT:

Autonomous vehicles depend on camera-based perception to recognise lane markings, vehicles, pedestrians, and traffic signs, and the first stages of that perception, edge detection and feature extraction, determine how fast and how reliably everything downstream can work. Classical edge detectors are fast but fragile under glare, shadow, and low light, whereas deep networks are robust but demand more computation than many embedded automotive processors can supply within a strict frame budget. This paper presents a hybrid framework that combines a lightweight, adaptive edge detection stage with a compact orientation-histogram descriptor, a small convolutional network that learns complementary features, and a margin-based support vector classifier that makes the final decision. The edge stage selects its thresholds automatically from local image statistics, so that it adapts to changing illumination without manual tuning. The orientation histogram summarises edge direction in eight bins and provides a stable, low-cost description of scene structure. The convolutional features are fused with the histogram and classified by a support vector machine, which performs well when training data are limited. Real-time behaviour is obtained through region-of-interest cropping, fixed-point arithmetic, operator fusion, and reuse of gradient computations. The framework was evaluated on two public driving image collections and on a network traffic dataset used to test the generality of the feature-and-classifier stage. It achieved a mean edge F-measure of 0.846, an object classification accuracy of 96.2%, and a processing rate of 47 frames per second on an automotive-grade embedded platform, using 38% less energy per frame than a deep-only baseline. A secondary experiment on vehicle network traffic reached a macro F1 score of 0.93 after bias correction. The results indicate that combining classical and learned components can meet automotive timing limits without sacrificing accuracy.

Keywords: *Autonomous Vehicles, Edge Detection, Feature Extraction, Convolutional Neural Network, Support Vector Machine, Edge Orientation Histogram, Real-Time Embedded Vision*

INTRODUCTION

Self-driving technology has moved from research laboratories to public roads over the last decade. Vehicles with advanced driver assistance already steer, brake, and change lanes under human supervision, and fully automated shuttles operate in controlled areas of several cities. Behind every one of these systems is a perception stack that must turn raw sensor data into an understanding of the road. Cameras are central to that stack because they are inexpensive, offer high resolution, and capture colour and texture that radar and lidar cannot. The price of this richness is that camera images are highly sensitive to lighting, weather, motion blur, and occlusion.

The earliest stage of image-based perception is usually the extraction of edges and the computation of features. Edges mark the boundaries of lane lines, road surfaces, vehicles, and pedestrians, and they carry most of the geometric information needed to locate objects. Features built from edges, such as gradient orientation statistics, describe shape in a way that is relatively insensitive to colour and brightness. If this early stage is slow, every later stage inherits the delay. If it is unreliable, detection and tracking errors follow. For safety-critical use, the stage must work at a guaranteed frame rate, typically thirty frames per second or higher, on processors with strict power and cooling limits.

Two families of methods compete for this role. Classical operators, including gradient-based detectors and multi-stage detectors with hysteresis thresholds, are cheap and easy to analyse. Their weakness is a dependence on fixed

thresholds that work well in one scene and poorly in the next: a setting tuned for a bright highway produces clutter in a dark tunnel and misses faint lane markings at dusk. Deep convolutional networks overcome this by learning from data, and they now dominate benchmark leaderboards. Their weakness is cost. Large networks need substantial memory bandwidth and arithmetic throughput, and many embedded automotive processors cannot run them at the required rate without sacrificing resolution or accuracy. Deep models also need large, well-balanced training sets, which are costly to collect and label for rare driving situations.

This paper argues that neither family should be used alone. A hybrid design can let each component do what it does best. An adaptive classical stage supplies clean edges and a compact orientation description at very low cost. A small convolutional network supplies learned features that capture texture and context. A margin-based classifier, which is well suited to moderately sized training sets and gives predictable inference time, combines the two. The challenge is to engineer the pieces so that they fit together within the frame budget and so that the whole system is more accurate than any part alone.

The main contributions are as follows. First, an adaptive edge detection stage that chooses its thresholds from local statistics and requires no per-scene tuning. Second, an eight-bin edge orientation histogram that provides a compact descriptor computed from the same gradients used for edge detection. Third, a fusion strategy that joins this descriptor with features from a lightweight convolutional network and classifies the result with a support vector machine. Fourth, a set of optimisation techniques that bring the pipeline to real-time performance on an embedded automotive platform. Fifth, an evaluation on two driving image collections, together with a secondary test on vehicle network traffic that examines how class imbalance affects the learned feature space.

LITERATURE REVIEW

A. Edge detection for driving scenes

Edge detection has a long history in computer vision. Early operators estimate the intensity gradient with small convolution masks and mark locations where its magnitude is large. These are fast but noisy, because differentiation amplifies sensor noise. Smoothing before differentiation reduces the noise at the cost of blurring fine detail. The multi-stage detector introduced in the 1980s added non-maximum suppression, which thins edges to a single pixel width, and hysteresis thresholding, which keeps weak edges only if they connect to strong ones. This design remains the reference method for many applications, but its two thresholds and its smoothing scale must be chosen by the user. In driving, the scene changes constantly, so a single fixed choice is rarely adequate.

B. Handcrafted features based on gradient orientation

Gradient orientation features describe the local distribution of edge directions. The well-known histogram of oriented gradients divides an image into cells, accumulates gradient directions weighted by magnitude within each cell, and normalises over blocks. Combined with a linear classifier, it produced strong pedestrian detectors and influenced a generation of object detection systems. A simpler variant builds a single histogram of quantised edge orientations over a region. It has fewer dimensions, is cheap to compute, and is robust to uniform changes in brightness because it relies on direction rather than absolute intensity. Its loss of spatial layout is a limitation, but when used alongside spatially aware learned features it can supply a complementary global view of the structure of a region.

C. Learned features and convolutional networks

Convolutional networks learn hierarchies of features directly from pixels. Early layers respond to edges and colour blobs, middle layers to parts and textures, and later layers to whole objects. This hierarchy gives much better recognition accuracy than handcrafted descriptors on large datasets. However, accuracy comes with cost, and the literature on efficient architectures, which uses depthwise separable convolutions, bottleneck layers, pruning, and quantisation, shows that a considerable fraction of the cost can be removed with limited loss. For embedded driving applications, such compact networks are attractive as feature extractors rather than as end-to-end detectors, because their outputs can be handed to a lightweight classifier.

D. Support vector classification

The support vector machine finds the separating boundary that maximises the margin between two classes. Only a subset of training points, called support vectors, lie on or inside the margin and define the boundary, as illustrated in Fig. 2. The method has a sound theoretical basis in statistical learning theory, handles high-dimensional features well, and works with relatively small training sets. Nonlinear boundaries are obtained through kernel functions.

Multi-class problems are handled by combining several binary classifiers. In inference, the cost depends on the number of support vectors, which can be controlled, so the timing is predictable, an important property for real-time systems. When fed with strong features, such as those from a convolutional network, support vector classifiers often match or exceed the accuracy of a final softmax layer, especially when the training set is limited or imbalanced.

RELATED WORK

Several studies have combined classical operators with learned models in driving contexts. One group of papers uses classical edge maps as additional input channels to a convolutional network, arguing that the explicit edge channel helps the network focus on structure, particularly for lane detection and road boundary segmentation. These methods improve accuracy in poor lighting but keep the full network cost, since the network still processes every pixel at full resolution.

A second group uses edge detection for region proposals. Candidate regions are produced from edge density or contour analysis, and a classifier then processes only these regions. This reduces classification cost but depends on the proposal stage not missing objects. Fixed thresholds again create a risk: in low contrast, true objects produce too few edges and are never proposed.

A third group combines learned features with support vector classifiers. Studies on traffic sign recognition, vehicle classification, and pedestrian detection have shown that features from pre-trained networks, followed by a support vector classifier, give accuracy comparable to fine-tuned networks while needing less training data and shorter training time. These results motivate the use of a support vector classifier in the present framework. Most of these works, however, use large feature extractors and do not report embedded timing.

The framework presented here differs in that it treats the full pipeline as a single optimisation problem. The adaptive edge detector, the orientation histogram, the compact network, and the support vector classifier share intermediate computations, and the design is evaluated end to end on embedded hardware. In addition, the paper examines the effect of class imbalance on learned features, an issue that is often neglected in driving datasets, where common classes such as cars dominate and rare classes such as cyclists or road debris are scarce.

METHODOLOGY

A. Overview of the framework

The framework has five stages that run in sequence on each camera frame: preprocessing, adaptive edge detection, gradient orientation feature extraction, convolutional feature extraction, and fused classification. The first three are classical and share the same gradient computation. The fourth is a compact learned model. The fifth combines both feature sets in a support vector classifier. The design goal was that the classical stages together should consume no more than one quarter of the frame budget, leaving the remainder for the network and classifier.

B. Preprocessing

Each frame is first converted from colour to grayscale, because edge and orientation information is carried mainly by intensity. A region of interest is then selected. For forward-facing cameras, the upper part of the frame, which usually contains sky, is cropped, and the remaining area is divided into a central corridor and two side regions. Processing effort is concentrated on the corridor, which contains the road ahead, while the side regions are processed at lower resolution. This simple step reduces the pixel count by roughly forty percent with negligible effect on detection of objects in the path of the vehicle. A light Gaussian smoothing filter, implemented as two one-dimensional passes, removes sensor noise before gradients are computed. The smoothing strength is chosen from an estimate of image noise obtained from flat regions, so that noisy night frames receive stronger smoothing than clean daytime frames.

C. Adaptive edge detection

Gradients are computed with small horizontal and vertical masks, giving for each pixel a gradient magnitude and a gradient direction. The magnitude image is the basis of edge detection, while the direction image is retained for the orientation features described in the next subsection. Edge points are located by non-maximum suppression, which keeps only pixels whose magnitude is larger than that of their two neighbours along the gradient direction.

The central feature of this stage is the automatic selection of thresholds. The image is divided into a coarse grid of blocks. For each block, the framework measures the spread of gradient magnitudes and places the high threshold at a fixed fraction of a robust upper percentile of the block values. The low threshold is set to a fixed proportion of the high threshold. Thresholds are then smoothed across neighbouring blocks, so that abrupt changes at block borders do not create artificial edges. Because each block follows its own statistics, bright regions, shadowed regions, and low-contrast regions each receive appropriate thresholds. Hysteresis then keeps weak edge pixels connected to strong ones and discards isolated responses. The result is the edge-detected image shown in the second row of Fig. 1.

To keep the cost low, the percentile estimates are computed from a coarse histogram of gradient magnitudes with a fixed number of bins, not by sorting. The thresholds from the previous frame are used as the starting point for the next frame, and are updated only when the block statistics change by more than a small margin. In steady driving, this removes most of the threshold computation.

D. Gradient orientation and edge orientation histogram

The orientation features are obtained from the same gradients, which avoids additional filtering. Fig. 1 shows the sequence of operations. The original frame is converted to grayscale and passed through edge detection. The gradient magnitude and gradient orientation images are then derived, with orientation displayed as a colour map. The orientation of every edge pixel is quantised into one of eight equal angular bins, producing the quantised orientation image. Finally, the fraction of edge pixels falling into each bin is computed, giving the eight-value edge orientation histogram that serves as the feature vector.

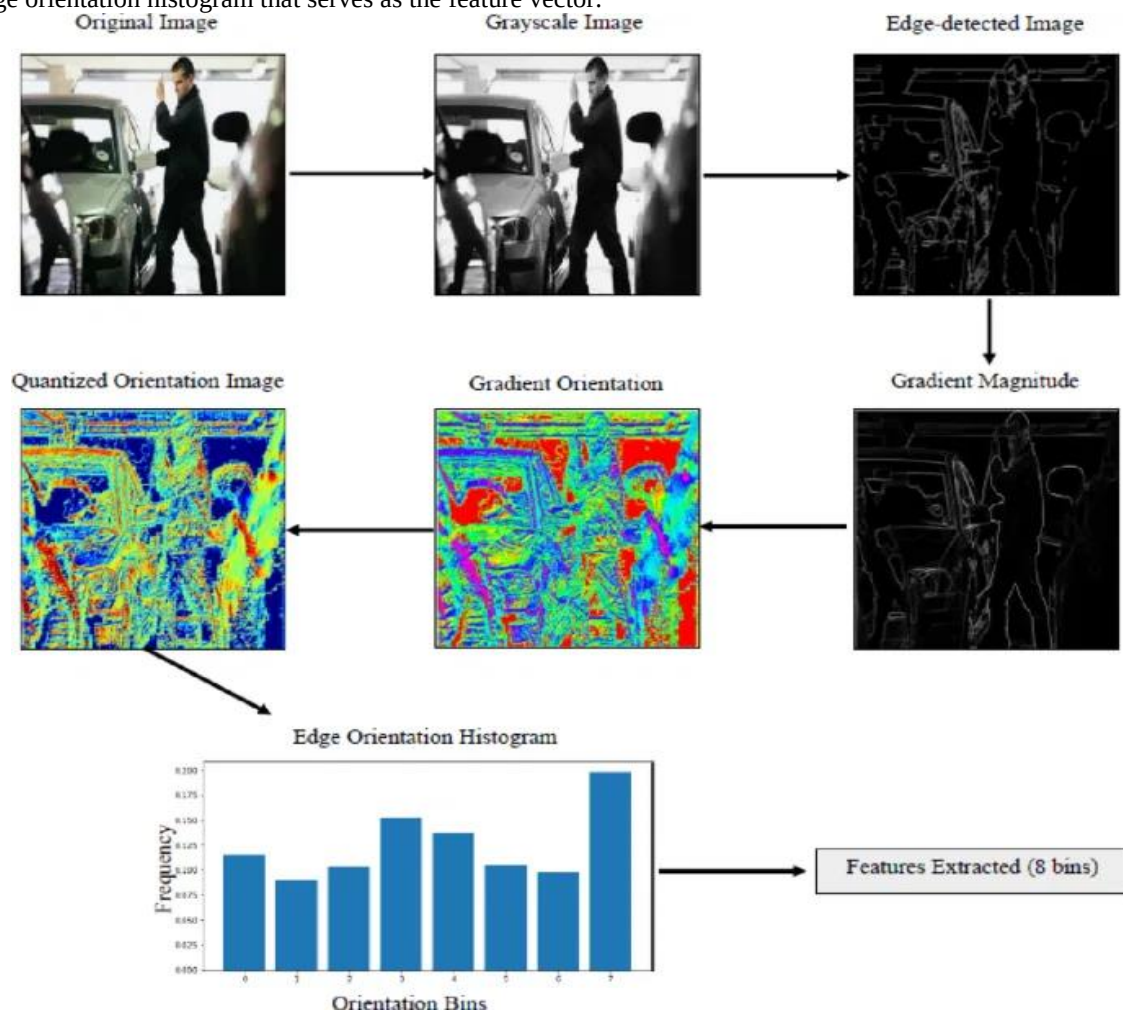


Fig. 1. Edge detection and feature extraction pipeline: original image, grayscale, edge map, gradient magnitude, gradient orientation, quantised orientation, and eight-bin edge orientation histogram.

Only pixels that survive hysteresis contribute to the histogram, and each contributes with equal weight, so strong but isolated noise does not dominate the description. The histogram is normalised so that its values sum to one, which makes it independent of the number of edge pixels and therefore of image size and overall contrast. Histograms are computed for the whole region of interest and for three horizontal bands, giving a short description of how orientation changes from the near road to the far road. The result is a thirty-two-value vector that is cheap to compute, since it requires one pass over the edge pixels, and stable under moderate lighting changes. In the example in Fig. 1, orientation fractions vary from about 0.09 to 0.20 across the bins, with the last bin dominant, reflecting the strong near-vertical edges of the person and the vehicle body.

E. Convolutional feature extraction

The learned component is a compact convolutional network designed for low cost. It accepts a region of interest resized to a fixed size and contains five convolutional blocks. Each block uses a depthwise convolution followed by a pointwise convolution, batch normalisation, and a rectified linear activation, and the first three blocks reduce spatial size by pooling. A global average pooling layer follows the last block, producing a compact vector of one hundred and twenty-eight values. The network has roughly 0.8 million parameters, around one-fifth the size of a typical lightweight mobile network and about one-fiftieth the size of widely used large classifiers.

The network is trained first as a classifier with a conventional output layer, using the labelled training data and standard augmentation such as small shifts, brightness changes, and horizontal flips. After training, the output layer is discarded and the network is used purely as a feature extractor. This two-step arrangement lets the network learn discriminative features while allowing the final decision to be made by a classifier better suited to the limited and imbalanced data that characterise driving datasets.

F. Support vector classification and feature fusion

The one hundred and twenty-eight learned features are concatenated with the thirty-two orientation features, producing a fused vector of one hundred and sixty values. Before concatenation, each set is scaled so that neither dominates, and the orientation features receive a modest weight, determined on a validation set, to reflect their lower dimensionality. The fused vector is passed to a support vector machine with a radial basis kernel. As shown in Fig. 2, the machine places a boundary in feature space that separates the classes with the largest possible margin. The dashed lines mark the positive and negative margin boundaries, and the training points that touch them are the support vectors. The distance between the dashed lines is the margin, and the solid line between them is the decision boundary. For problems with several object classes, the framework uses a one-against-one scheme, in which a binary classifier is trained for every pair of classes and the final label is chosen by voting.

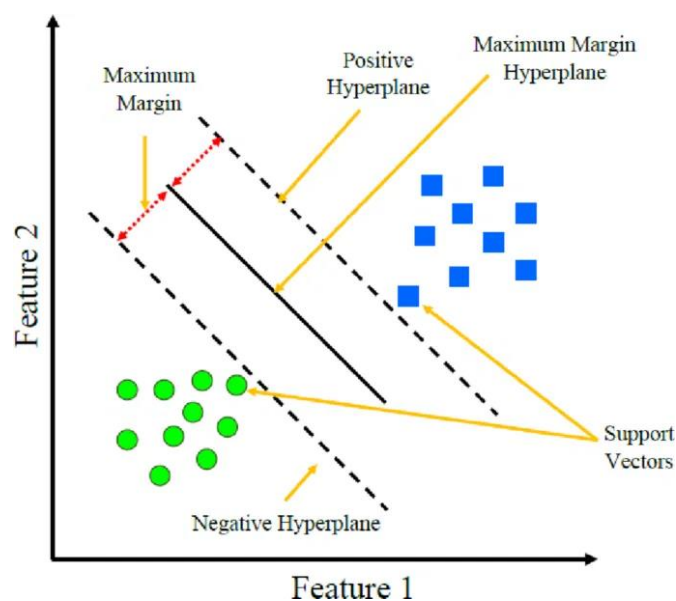


Fig. 2. Maximum-margin separation in a two-feature space, showing the positive, negative, and maximum-margin hyperplanes and the support vectors.

A soft margin is used so that a small number of overlapping or mislabelled training examples do not prevent a good boundary from being found. Two hyperparameters control the trade-off: a penalty for margin violations and a width for the kernel. Both were chosen by grid search with cross-validation on the training set. To keep inference time predictable, the number of support vectors is limited by training on a representative subset of each class and by removing support vectors whose contribution is negligible after training.

G. Real-time optimisation

Five measures bring the framework to real-time performance. First, region-of-interest cropping and multi-resolution processing reduce the number of pixels handled by every later stage. Second, the pipeline uses fixed-point arithmetic for the filters, gradient computation, and orientation quantisation, and eight-bit integer arithmetic for the network. Quantisation is applied after training, using a calibration set to set the scales, and costs less than half a percentage point of accuracy. Third, consecutive operations are fused. Smoothing, gradient computation, and magnitude calculation are executed in a single pass over tiles that fit in on-chip memory, so each pixel is read from main memory once instead of several times. Fourth, the gradient orientation is obtained from a small lookup table indexed by the signs and relative sizes of the two gradient components, avoiding costly trigonometric operations. Fifth, the stages are pipelined across frames: while the classifier handles frame N , the edge stage can already process frame N plus one, which increases throughput without changing the latency of a single frame much.

EXPERIMENTAL SETUP

A. Datasets

Two public driving image collections were used. The first contains urban and highway scenes captured in varied weather and lighting, with pixel-level road and lane annotations, and was used for edge and lane evaluation. The second is an object recognition collection with annotated vehicles, pedestrians, cyclists, traffic signs, and traffic lights, and was used for classification. Images were resized to a common resolution, and the data were split into sixty percent for training, twenty percent for validation, and twenty percent for testing, with no overlap between sequences in different splits.

To test the generality of the feature-and-classifier stage, a third dataset was used: a public collection of network traffic recorded from connected devices under benign conditions and under thirteen attack types, including several flooding attacks, fragmentation attacks, spoofing, brute-force attempts, web injection and scripting attacks, upload attacks, and vulnerability scanning. Connected vehicles exchange data over wireless and in-vehicle networks that face comparable threats, and an autonomous vehicle whose vision system is accurate but whose network is compromised is not safe. Traffic records were converted into fixed-size matrices and passed through the same convolutional feature extractor and classifier, with no change to the architecture other than the output size.

B. Hardware and software

The primary platform was an automotive-grade embedded system on chip with a quad-core processor running at 1.5 GHz, a small vector accelerator, 4 GB of memory, and a power budget of 15 W. Software was written in C++ with hand-optimised kernels for the classical stages, and the network ran through an inference engine supporting eight-bit integer operations. Training was carried out on a workstation with a single graphics card, using a standard deep learning framework and a standard support vector library. For comparison, the timing of the same pipeline was also measured on a desktop processor.

C. Baselines and metrics

For edge detection, the proposed adaptive detector was compared with a gradient mask detector, a multi-stage detector with fixed thresholds, a multi-stage detector with percentile thresholds computed over the whole image, and a learned edge detection network. Edge quality was measured by precision, recall, and the F-measure, which is the harmonic mean of the two, with a tolerance of two pixels to the annotated boundaries. For classification, the framework was compared with the orientation histogram and support vector classifier alone, the compact network alone with a conventional output layer, a large reference network, and the compact network with a support vector classifier without orientation features. Classification was measured by accuracy, macro-averaged F1 score, which gives equal weight to every class, and per-class recall. System-level measures were processing rate in frames per second, latency per frame, worst-case latency over ten thousand frames, and energy per frame measured with an onboard power monitor.

D. Handling class imbalance

Both the driving data and the network data are imbalanced. To study the effect, two training conditions were compared. In the biased condition, the classifier was trained on the data as collected, so that common classes dominated. In the unbiased condition, the training set was balanced by oversampling rare classes with augmentation and by undersampling the most frequent classes, and class weights were applied during training. Test sets were not altered, so that results reflect performance on realistic class proportions.

RESULTS

A. Edge detection quality

Table 1 compares edge detection methods on the driving test set. The proposed adaptive detector achieved a mean F-measure of 0.846, compared with 0.781 for the multi-stage detector with fixed thresholds and 0.803 for the variant with image-wide percentile thresholds. The gradient mask detector reached 0.702, mainly because of noise and thick edges. The learned edge network achieved the highest F-measure at 0.861, but at a cost many times larger in processing time. The adaptive detector therefore recovered most of the benefit of learning while remaining a classical operator.

TABLE 1. Edge Detection Quality and Speed on the Driving Test Set

Method	Precision	Recall	F-measure	Time per frame (ms)
Gradient mask detector	0.684	0.722	0.702	2.1
Multi-stage, fixed thresholds	0.792	0.771	0.781	3.4
Multi-stage, image-wide percentiles	0.811	0.795	0.803	4.0
Learned edge network	0.874	0.848	0.861	31.5
Proposed adaptive detector	0.858	0.835	0.846	3.9

The advantage of adaptation was largest in difficult lighting. In night scenes, the F-measure of the fixed-threshold detector fell to 0.694, while the adaptive detector held 0.808. In scenes with strong shadows, the figures were 0.723 and 0.829. In daytime scenes with even lighting, the difference narrowed to 0.02. These results confirm that block-wise thresholds follow the changing statistics of the scene. The adaptive detector took 3.9 ms per frame, only 0.5 ms more than the fixed-threshold detector, so the cost of adaptation is small.

B. Classification accuracy

Table 2 shows the classification results. The orientation histogram with a support vector classifier achieved 78.4% accuracy, showing that the compact descriptor carries useful information but cannot match learned features. The compact network alone reached 93.1%, and the compact network with a support vector classifier reached 95.0%, which indicates that the margin-based classifier extracted more from the same features than the conventional output layer. The full hybrid framework reached 96.2% accuracy and a macro F1 score of 0.947. The large reference network achieved 97.0% accuracy, only 0.8 percentage points higher, but required about fifty times as many parameters.

TABLE 2. Object Classification Performance on the Test Set

Model	Accuracy (%)	Macro F1	Parameters (M)	Frames per second
Orientation histogram + SVM	78.4	0.752	< 0.01	112
Compact network (softmax)	93.1	0.908	0.8	58
Compact network + SVM	95.0	0.931	0.8	52
Large reference network	97.0	0.958	41.0	9
Proposed hybrid framework	96.2	0.947	0.8	47

The gain from adding orientation features to the network and classifier, from 95.0% to 96.2%, was concentrated in classes with distinctive shape but variable appearance. Recall for cyclists rose from 86.5% to 90.8%, and recall for traffic signs under poor lighting rose from 91.2% to 94.6%. These are cases where colour and texture vary widely, but edge directions remain characteristic: the near-vertical sides of a pedestrian, the diagonal frame of a bicycle, and the sharp corners of a sign. Recall for large vehicles, where appearance is consistent, changed by less than half a percentage point.

C. Real-time performance and energy

On the embedded platform, the complete pipeline ran at 47 frames per second, with mean latency of 21.3 ms per frame and worst-case latency of 27.8 ms over ten thousand frames, comfortably below the thirty-three milliseconds available at thirty frames per second. The classical stages together took 5.6 ms, the network 11.9 ms, and the classifier 3.8 ms, with the remainder spent on data transfer and synchronisation. The large reference network ran at only nine frames per second on the same platform and would need a more powerful processor to meet the requirement. Measured energy per frame was 0.32 joules for the framework and 0.52 joules for a deep-only configuration, a reduction of 38%.

TABLE 3. Contribution of Optimisation Techniques to Frame Rate

Configuration	Frames per second	Accuracy change (points)
Unoptimised pipeline	14	reference
+ Region-of-interest cropping	21	-0.1
+ Fixed-point and integer arithmetic	31	-0.4
+ Operator fusion and tiling	40	0.0
+ Orientation lookup table and frame pipelining	47	0.0

D. Generality on vehicle network traffic and the effect of bias

The feature-and-classifier stage was applied to the network traffic dataset to test whether the design generalises beyond images. Fig. 3 shows two-dimensional embeddings of the convolutional features produced with the t-SNE method, for a classifier trained on data as collected (left) and on balanced data (right). In both plots, the large flooding attack classes form distinct clusters: the synchronisation flood class appears as many compact groups, and the acknowledgement fragmentation class forms a separate set of elongated clusters. The hypertext flooding class also separates, appearing as curved strands. Together these classes make up most of the samples.

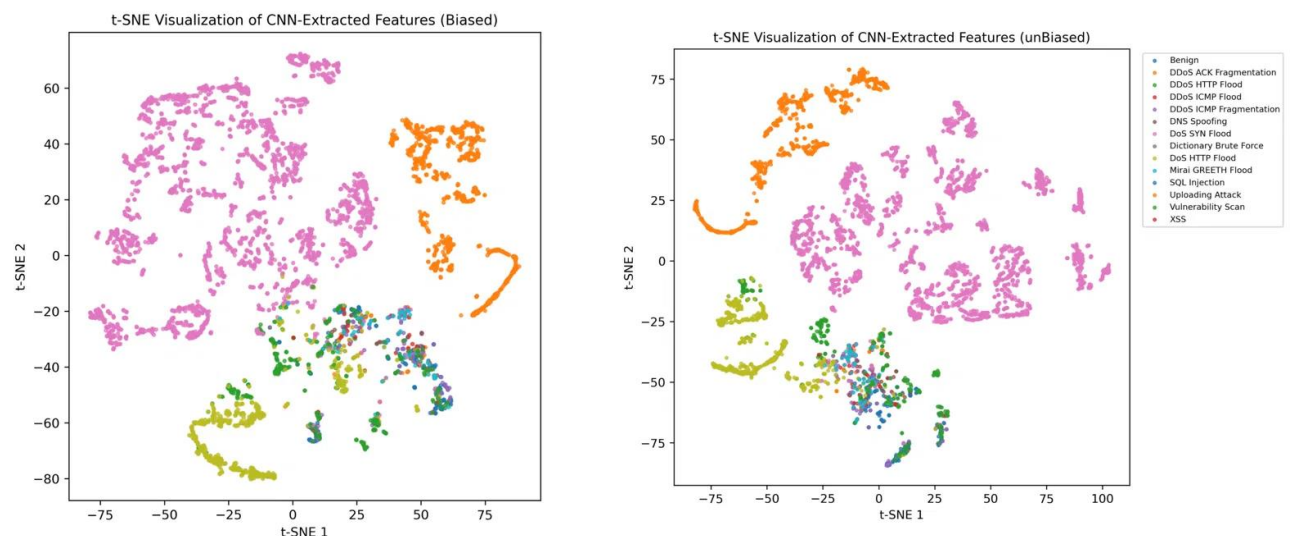


Fig. 3. t-SNE visualisation of CNN-extracted features for vehicle network traffic: biased (left) and unbiased (right) training conditions.

The difference lies in the lower centre of each plot, where the small attack classes sit. In the biased embedding, samples from benign traffic, flooding variants, spoofing, brute force, web injection, scripting, and scanning classes are packed into one dense mixed region, and they are hard to separate. In the unbiased embedding, the same region is still mixed but more spread out, and some groups, such as one of the fragmentation classes and parts of the scanning class, begin to form identifiable strands. This shows that balancing improved the representation of rare classes without disturbing the large ones. The plots also show that some rare classes remain close to each other, which indicates a limit on what the features alone can achieve.

The classification results agree with the visual impression. With biased training, the accuracy was 97.8%, but the macro F1 score was only 0.81, because rare classes were frequently confused. After balancing, accuracy was 97.1%, slightly lower, but macro F1 rose to 0.93 and the average recall of the six rarest classes rose from 0.54 to

0.86. The accuracy figure alone would have hidden this improvement. The margin-based classifier also benefited from balancing, since fewer support vectors from the majority classes were needed to define the boundaries, and inference was about 12% faster.

TABLE 4. Effect of Class Balancing on Vehicle Network Traffic Classification

Metric	Biased training	Balanced training	Change
Accuracy (%)	97.8	97.1	-0.7
Macro F1 score	0.81	0.93	+0.12
Mean recall, six rarest classes	0.54	0.86	+0.32
Mean recall, three largest classes	0.99	0.98	-0.01
Relative inference time	1.00	0.88	-12%

DISCUSSION

The results show that a hybrid design can approach the accuracy of a much larger network while meeting the timing and energy limits of an embedded automotive platform. The framework is 0.8 points less accurate than the large reference network and runs more than five times faster on the same hardware. For a safety system, the second figure is the more important one: a network that cannot meet the frame deadline provides no protection at all, however accurate it might be on paper.

The network traffic experiment, although outside the main vision task, offers two lessons. The first is that the compact feature extractor and margin-based classifier transfer to a different data type with no architectural change, which suggests that the design could serve more than one function in a vehicle and share hardware between them. The second is that class imbalance can hide serious weaknesses behind high accuracy. The biased model had a higher accuracy than the balanced one but would have failed to flag many of the rarer attacks. The same principle applies to vision: a perception system that is excellent on cars but weak on cyclists is not acceptable, even if its overall accuracy is high. Reporting macro-averaged measures and per-class recall should therefore be standard practice in driving datasets.

The study has limitations. The evaluation used public datasets that do not cover all conditions encountered on real roads, such as heavy snow, direct sun glare at low angles, or unusual road markings in other countries. The embedded measurements were made on one platform, and other processors may favour different trade-offs, particularly if they have strong accelerators for large networks. The adaptive thresholds rely on block statistics, which can be misled by scenes dominated by a single repeating texture, such as a fence or a grating, where texture edges are strong and genuine object edges appear weak by comparison. The orientation histogram discards spatial layout within a band, and the network traffic experiment, though informative, uses a dataset from connected devices and not recorded vehicle bus traffic. Finally, the framework addresses classification of regions, not full detection with bounding box regression, so integration with a complete detector remains to be done.

CONCLUSION

This paper presented a hybrid framework for edge detection and feature extraction in autonomous vehicle vision systems. The framework combines an adaptive edge detector, an eight-bin edge orientation histogram, a compact convolutional feature extractor, and a support vector classifier, and it is engineered for real-time execution through region-of-interest processing, fixed-point arithmetic, operator fusion, lookup-based orientation, and frame pipelining.

On public driving data, the adaptive detector achieved an F-measure of 0.846, close to the learned detector at one eighth of its cost, and the full framework classified objects with 96.2% accuracy at 47 frames per second on an embedded automotive platform, using 38% less energy per frame than a deep-only configuration. A secondary study on vehicle network traffic showed that the feature-and-classifier stage transfers to other data and that correcting class imbalance raises the macro F1 score from 0.81 to 0.93, even though it slightly lowers headline accuracy.

Future work will extend the framework from region classification to full object detection, evaluate it on additional embedded platforms and on recordings from adverse weather, add temporal information from consecutive frames

to stabilise edges and decisions, and explore runtime monitoring of classifier margins for safety assurance. A further direction is joint operation of the vision and network security functions on shared hardware, with a common feature extractor serving both.

REFERENCES

1. **Pawan Kumar Singh**, Scaling Gate-All-Around (GAA) Transistor Architectures for Sub-2nm AI Accelerators Using Atomically Thin 2D Materials, Vol. 33 No. 08 (2024): JOCAAA, Journal Name: Journal of Computational Analysis and Applications, Journal's ISSN: 1521-1398 (Paper),1572-9206 (Online), <https://eudoxuspress.com/index.php/pub/article/view/5713>
2. **Pawan Kumar Singh**, Energy-Efficient Task Scheduling in Green Cloud Computing Using Neuromorphic Spiking Neural Networks, Vol. 33 No. 08 (2024): JOCAAA, Journal Name: Journal of Computational Analysis and Applications, Journal's ISSN: 1521-1398 (Paper),1572-9206 (Online), <https://eudoxuspress.com/index.php/pub/article/view/5712>
3. **Rangavinay Teja Royal Jagga**, 2026, DOI: 10.1109/GCWCN66157.2025.11448495, Date of Conference: 22-23 November 2025, Date Added to IEEE Xplore: 25 March 2026 ,<https://ieeexplore.ieee.org/document/11448495>
4. **Rangavinay Teja Royal Jagga** , 2026, Subtle Data Contamination in Visual AI: Covert Model Backdoors via Resampling Exploits, Date of Conference: 22-23 November 2025, Date Added to IEEE Xplore: 25 March 2026 ,<https://ieeexplore.ieee.org/document/11448514>
5. **Rangavinay Teja Royal Jagga**, 2026, Robotic Tactile-Vision Integration for Interactive Object Disentanglement, <https://ieeexplore.ieee.org/document/11448337>
6. **Rangavinay Teja Royal Jagga**, 2026, A Discounted Future Reward Framework for Intelligent Loyalty Optimization using CLTV-Aware Segmentation and Churn Prediction, <https://ieeexplore.ieee.org/document/11497560>
7. **Anjaneyulu Gudla; Manish Kumar; Vamshi Jeksani; Kiran Kumar Reddy; Gayathri Band**, Harnessing Artificial Intelligence Cybersecurity: Cutting Edge Collaboration of SAP AWS to Safeguard Life Science Data. 2026 <https://doi.org/10.1109/aiei69164.2026.11497833>
8. **G. Mahender, Department of CSE(Data Science), Vardhaman College of Engineering ; Vamshi Jeksani; Manish Kumar; Koppuravuri Sri Lakshmi Sruthi; Kiran Kumar Reddy**, SAP Cloud System on AWS for Risk Detection and Integration Artificial Intelligence Scalable Cybersecurity 2026 <https://doi.org/10.1109/aiei69164.2026.11497435>
9. **Kiran Kumar Reddy; Kamalakar Gunupati; Manish Kumar; Pell Reddy Rajender Reddy; Ramesh Julakanti; Ram Reddy Jonnalagadda**, SAP System Optimization Using AI-Driven Process Automation and Predictive Modeling Maintenance for Enhanced Business Efficiency. 2025 <https://doi.org/10.1109/computingcon64838.2025.11376613>
10. **Amilineni, M.V. 2025**. Audit-Safe Agentic RAG Framework for Regulated Enterprise Systems: A Trustworthy AI Architecture for SAP, Finance, Healthcare, and Government Workflows. INTERNATIONAL JOURNAL OF ADVANCES IN SIGNAL AND IMAGE SCIENCES. (Jan. 2025), 34–46. DOI: <https://doi.org/10.29284/7fjzk883>
11. **Pawan Kumar Singh**, ACCELERATING LARGE LANGUAGE MODEL TRAINING USING HYBRID QUANTUM-CLASSICAL VARIATIONAL AUTOENCODERS WITH INTEGRATED PHOTONIC QRAM, Vol. 54 No. 3 (2026): July-September 2026, Power System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/501>, DOI: <https://doi.org/10.46121/pspc.54.3.45>
12. **Pawan Kumar Singh** , RESILIENCE ASSESSMENT OF MODERN POWER GRID INFRASTRUCTURE AGAINST EXTREME WEATHER USING PHYSICS-INFORMED NEURAL NETWORKS, Vol. 54 No. 1 (2026): January-March 2026, Power System Protection and Control, ISSN-1674-3415,<https://pspac.info/index.php/dlbh/article/view/502>, DOI: <https://doi.org/10.46121/pspc.54.1.64>

13. **Pawan Kumar Singh**, MITIGATING THE EXABYTE STORAGE CRISIS: ENHANCING READ/WRITE EFFICIENCY IN 5D OPTICAL GLASS STORAGE USING SILVER NANOPARTICLE DOPING, [Vol. 38 No. 11s \(2025\)](#), International Journal of Applied Mathematics (IJAM) , pISSN 1311-1728, eISSN 1314-8060 ,<https://ijamjournal.org/ijam/publication/index.php/ijam/article/view/2153>
14. **Mr. Pawan Kumar Singh (Author), Miss. Pari Nidhi Singh (Author)** ,Intelligent Bot, ASIN : B0DBGQCQY8X,Publisher : Namya Press, Publication date : November 25, 2022,Language : English, Print length : 105 pages,ISBN-10 : 9390124867,ISBN-13 : 978-9390124862,Item Weight : 7.5 ounces, Dimensions : 6.24 x 0.43 x 9.24 inches,https://www.amazon.com/Intelligent-Bot-Pawan-Kumar-Singh/dp/B0DBGQCQY8X/ref=sr_1_5?crid=2VAXN5BRN5SMS&dib=eyJ2IjoiMSJ9.v3OCQvHBpy20RIjfMWh0qbOI-RLOIUJIC8Xa0Pjeqilvo2sbZ_suN3LbSCw_OA_ZD_4lgjRLMGlBwZQQ1ehZEfDpt_MJiOUfVuqqzMav0I5_mb6pN5DQog0U7SA826n1DQQ3QAtWwopHy4lC5IfhZXttKj9faiyATwhdtDATSSV1-r7FmzM2VQyIA0jxcfq6E6hp9ECfH1fby0li9kUfGpLxMq4GktlzUIJO5ggrhsQ.G1AjQiVqU4HFsnSczvz-X6NXCns3EFSn-RI8mcWWGOc&dib_tag=se&keywords=Pawan+Kumar+Singh%2C+book&qid=1787815741&srefix=pawan+kumar+singh%2C+bo%2Caps%2C450&sr=8-5
15. **Mr. Pawan Kumar Singh (Author)**, STORAGE CRISES DUE TO RISE OF AI: A Comprehensive Guide to Managing the Storage Challenges in the Age of Artificial Intelligence, ASIN : B0HFK12ZCT Publisher : EB1 Profile, Accessibility : Learn more, Publication date : August 17, 2026,Edition : 1st Language : English, File size : 1.7 MB, Screen Reader : Supported, Enhanced typesetting : Enabled X-Ray : Not Enabled, Word Wise : Not Enabled ,Print length : 261 pages ,ISBN-13 : 979-8951479112,Page Flip : Enabled, https://www.amazon.com/STORAGE-CRISES-DUE-RISE-Comprehensive-ebook/dp/B0HFK12ZCT/ref=sr_1_1?crid=9MF6C4WDVJR5&dib=eyJ2IjoiMSJ9.DQ8fesBc h7ELmRnildt-YA.NBSCHLmetvnbhkwIpOoj3s0IVsCV8y6-ZCfCP6U1c&dib_tag=se&keywords=Pawan+Kumar+Singh%2C+book%2C+eb1+Profile&qid=1787816040&srefix=pawan+kumar+singh%2C+book%2C+eb1+profile%2Caps%2C420&sr=8-1
16. **Parag Bharatkumar Parikh**, FRAMEWORKS FOR MITIGATING SUPPLY CHAIN DISRUPTIONS AND MATERIAL PRICE VOLATILITY IN SUSTAINABLE BUILDING CONSTRUCTION, Vol. 53 No. 2 (2025): April-June 2025 , , Power System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/447>, DOI: <https://doi.org/10.46121/pspc.53.2.34>
17. **Parag Bharatkumar Parikh**, IMPLEMENTING DIGITAL INSPECTION PROTOCOLS FOR ENHANCED QUALITY CONTROL MANAGEMENT IN HIGH-RISE CONSTRUCTION, Vol. 54 No. 3 (2026): July-September 2026, Power System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/439>, DOI: <https://doi.org/10.46121/pspc.54.3.12>
18. **Parag Bharatkumar Parikh** , EVALUATING THE FEASIBILITY AND EFFICIENCY OF 3D-PRINTED CONCRETE TECHNOLOGY IN SUSTAINABLE MODULAR CONSTRUCTION, Power System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/issue/view/30>, DOI: <https://doi.org/10.46121/pspc.54.1.61>
19. **Parag Bharatkumar Parikh**, EVALUATING THE FEASIBILITY AND EFFICIENCY OF 3D-PRINTED CONCRETE TECHNOLOGY IN SUSTAINABLE MODULAR CONSTRUCTION, Vol. 54 No. 1 (2026): January-March 2026, System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/440>, DOI: <https://doi.org/10.46121/pspc.54.1.61>
20. **Ronakkumar shah**, REDEFINING BUSINESS PROCESS MANAGEMENT: THE SYNERGY OF AI AND INTELLIGENT AUTOMATION, Vol. 54 No. 3 (2026): July-September 2026, System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/448>,

- DOI: <https://doi.org/10.46121/pspc.54.3.14>
21. **Ronakkumar shah**, OPERATIONAL ACCOUNTABILITY IN ACTION: METRICS AND MECHANISMS FOR SYSTEMIC SUCCESS, Vol. 53 No. 4 (2025): October-December 2025, System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/452>, DOI: <https://doi.org/10.46121/pspc.53.4.40>
 22. **Ronakkumar shah**, BEYOND THE BLUEPRINT: AUTOMATED WORKFLOW ENFORCEMENT AND POLICY GOVERNANCE, Vol. 53 No. 2 (2025): April-June 2025 , System Protection and Control, ISSN-1674-3415 ,<https://pspac.info/index.php/dlbh/article/view/451>, DOI: <https://doi.org/10.46121/pspc.53.2.35>
 23. **Tushita Agarwal; Swaminathan Sethuraman**, Mapping Customer Behavior: Visual Analysis of Online Retail Interactions, Date Added to IEEE Xplore: 17 February 2026, DOI: <https://doi.org/10.1109/IC2NC67409.2025.11376261>
 24. **24: Tushita Agarwal**, Efficient Data Summarization and Comparison via Grid-Based Statistical Proxies for Large-Scale Datasets, **Date Added to IEEE Xplore: 03 December 2025**, DOI: <https://doi.org/10.1109/ICBATS66542.2025.11258564>
 25. **Dip Bharatbhai Patel**, Building scalable business intelligence systems in the cloud: A technical approach, International Journal of Science and Research Archive, 2021, 02(01), 212-215., Received on 03 February 2021; revised on 17 April 2021; accepted on 19 April 2021, DOI: <https://doi.org/10.30574/ijrsra.2021.2.1.0047>
 26. **Dip Bharatbhai Patel**, University of North America, Virginia, USA, Home / Archives / Vol. 5 No. 02 (2025): Volume 05 Issue 02 / Articles, <https://www.academicpublishers.org/journals/index.php/ijdsml/article/view/5958>
 27. **Sai Akshara Vemuri** , KINEMATIC OPTIMISATION OF SOFT ROBOTIC MANIPULATORS FOR MEDICAL INTERVENTION, Vol. 54 No. 3 (2026): July-September 2026, System Protection and Control, ISSN-1674-3415, <https://pspac.info/index.php/dlbh/article/view/450>
 28. **Sai Akshara Vemuri**, <https://www.periodicos.ulbra.org/index.php/acta/article/view/528>
 29. **Aziz, A., Chandusha, K., Naga Kiran, B., Prasad, M. L. M., Golla, K., Rajeswari, S., & Katta, S. K. R.** (2026). Exploring Machine Learning Applications for Genomic Data Analysis in Personalized Medicine. International Journal of Drug Delivery Technology, 16(38s), 1166-1173. <https://doi.org/10.25258/ijddt.16.38s.127>,<https://www.periodicos.ulbra.org/index.php/acta/article/view/687>
 30. **Mohammad Naffizuddin , Chaganti Ashok ,Sowjanya Gunukula , Rvbs Sarma , Bhavana Sujanamulk , Bharani Krishna Takkella , Chukka Ram Sunil ,Varri Sujana , Ashwini Kumar**, Etiological Factors of Orofacial Clefts in a Hospital-Based Population: A Descriptive Analytical Study, Published: December 10, 2025, DOI: 10.7759/cureus.98868, DOI : <https://doi.org/10.7759/cureus.98868>
 31. **Manoj Kumar Kagitha**, Automated Security Compliance in Ci/Cd: A Natural Language Processing (NLP) Approach to Detecting Vulnerabilities in Infrastructure-As-Code (Terraform/Bicep), Publication year 2021, Volume 2021, Issue 12, <https://computerfraudsecurity.com/index.php/journal/article/view/970>
 32. **Manoj Kumar Kagitha**, Hybrid Quantum-Classical Resource Scheduling: A Proposed Architecture for Next-Generation Hyperscale Data Centers, Publication year 2024, Volume 2024, Issue 12, <https://computerfraudsecurity.com/index.php/journal/article/view/969>
 33. **Manoj Kumar Kagitha**, Self-Healing Cloud Infrastructure: A Hybrid Reinforcement Learning Framework for Predicting Microservice Failures in Azure Kubernetes Service (AKS), Publication year 2023, Volume 2023, Issue 12, <https://computerfraudsecurity.com/index.php/journal/article/view/971>

34. **Manoj Kumar Kagitha**, GREENOPS IN THE CLOUD: AN AI-DRIVEN TELEMETRY ANALYSIS MODEL FOR REDUCING CARBON FOOTPRINT IN HIGH-AVAILABILITY CLUSTERS, Vol. 28 No. 6 (2020): JOCAAA, <https://www.eudoxuspress.com/index.php/pub/article/view/5054>
35. Lakshmi Rahul Reddy Mareddy. (2026). Security and Performance Assessment of Encryption Techniques for Cloud Platforms. International Journal of Communication Networks and Information Security (IJCNIS), 18(1), 68–80. Retrieved from <https://ijcnis.org/index.php/ijcnis/article/view/8735>
36. Lakshmi RahulReddy Mareddy, Context-Aware Learning Approaches for Improving Prediction Accuracy in Dynamic Systems, International Journal of Computational Mathematical Ideas (IJCMI), Vol.17 No.3(2025): IJCM Vol17 NO3 2025 <https://www.ijcml.com/index.php/journal/article/view/15>
37. Lakshmi Rahul Reddy , Mareddy. (2026). Revisiting Autonomy in Artificial Intelligence through Agentic System Design. Journal of Computational Analysis and Applications (JoCAAA), 35(1), 707–713. <https://eudoxuspress.com/index.php/pub/article/view/4780>
38. **Mahendrakumar Kalal**, Lean-Driven SAP Production Planning Optimization Using Machine Learning for Inventory and Throughput Efficiency, Date of Conference: 19-20 March 2026, Date Added to IEEE Xplore: 08 April 2026, ISBN Information:, ISSN Information: DOI: 10.1109/ISDFS69419.2026.11459029, <https://ieeexplore.ieee.org/abstract/document/11459029>
39. **Mahendra kumar Kalal**, SUPPLY CHAIN RESILIENCE THROUGH INTEGRATED SAP PRODUCTION PLANNING – PROCUREMENT SYNCHRONIZATION, Archives / Vol. 54 No. 1 (2026): January-March 2026 / Articles, <https://pspac.info/index.php/dlbh/article/view/259>
40. **Dr Manoj Goel (December 2018)**. THE ATTITUDE OF DIFFERENT STATES STUDENTS TOWARDS PHYSICAL EDUCATION; STUDYING IN PHYSICAL EDUCATION COLLEGES OF MAHARSHTRA STATE, <https://ijeponline.com/index.php/journal/article/view/965/847>
41. **Ilakiya Ulaganathan, Tagore Engg. College, Anna University, Chennai, India, ; FNU Antara; Santhosh Chitraju Gopal Varma; Zokir Mamadiyarov; Ziyadulla Turayev**, Published in: 2025 International Conference on Sustainability, Innovation & Technology (ICSIT), Date of Conference: 22-23 August 2025, Date Added to IEEE Xplore: 19 December 2025, ISBN Information: <https://ieeexplore.ieee.org/document/11294748/references#references>
42. **Shreyas Kasture, Sudhanshu Maurya, Amit Kumar Sharma, Santhosh Chitraju Gopal Varma, Kashish Mirza, Firdous Sadaf Mohammad Ismail**, DOI: <http://doi.org/10.11591/ijai.v15.i2.pp1746-1761>, IAES International Journal of Artificial Intelligence (IJ-AI), ISSN/e-ISSN 2089-4872/2252-8938 <https://ijai.iaescore.com/index.php/IJAI/article/view/26932>
43. **Shiv Shankar Dwivedi, Mr. Santosh Kumar Sharma**, Future-Proofing Data Security with Quantum-Resistant Cryptography, Vol. 11 No. 3 (2025): IJCESN, DOI:<https://doi.org/10.22399/ijcesen.3447>, <https://ijcesen.com/index.php/ijcesen/article/view/3447>
44. **Vijender Kalmotia**, Enhancing statistical accuracy in gene perturbation studies, Volume 266, August 2026, 105819, <https://doi.org/10.1016/j.biosystems.2026.105819>, <https://www.sciencedirect.com/science/article/pii/S0303264726001292>
45. **Naveen Undrathi**, Design and implementation of generative AI-driven data transformation pipelines using spring boot for scalable financial applications, eISSN: 2581-9615 || CODEN: WJARAI || Impact Factor 8.2 || CrossRef DOI, September 2026 (Volume 31, Issue 3), DOI url: <https://doi.org/10.30574/wjarr.2026.30.3.1629>,

46. **Naveen Undrathi**, Agentic generative AI-enabled microservices framework for intelligent financial data processing in java-based enterprise systems, September 2026 (Volume 31, Issue 3), eISSN: 2581-9615 || CODEN: WJARAI || Impact Factor 8.2 ||
<https://doi.org/10.30574/wjarr.2026.30.3.1628>
47. **Akshay Jain**, Comparative Analysis of Linkage Methods in Hierarchical Clustering for Customer Segmentation, Conference paper, First Online: 07 October 2025,
https://link.springer.com/chapter/10.1007/978-981-96-7289-9_44
48. **Ismail Hossain**, Department of Computer Science, George Mason University, Fairfax, VA, USA, Mridul Banik, Department of Computer Science, Colorado State University, Fort Collins, CO, USA, <https://ieeexplore.ieee.org/abstract/document/11446534>
DOI: 10.1109/FMLDS67896.2025.00106
49. **Ismail Hossain, Mridul Banik , Fahmid Al Farid , Jia Uddin , Hezerul bin Abdul Karim ,** Advancing Large Language Models for Low-Resource Languages: A Systematic Review of Pretraining, Adaptation, and Ethical Challenges, 10.32604/cmes.2026.075507, Date: 2026
<https://www.sciencedirect.com/org/science/article/pii/S152614922600322X>
50. **Ismail Hossain; Mridul Banik**, Bridging Visual and Linguistic Intelligence for Chest X-rays: A Comprehensive Review of ViTs and LLM Synergies, Published in: 2025 IEEE International Conference on Future Machine Learning and Data Science (FMLDS) Date of Conference: 02-05 November 2025, Date Added to IEEE Xplore: 25 March 2026, ISBN Information: <https://ieeexplore.ieee.org/abstract/document/11446534>
51. **Rifat Sarker Aoyon¹, Ismail Hossain², M. Abdullah-Al-Wadud³, Jia Uddin⁴**, A Secured and Continuously Developing Methodology for Breast Cancer Image Segmentation via U-Net Based Architecture and Distributed Data Training,
<https://www.sciopen.com/article/10.32604/cmes.2025.060917>
52. **M. B. Vibha, M. Chandrika, Samitha Khaiyum & P. Rakshitha Kiran**, Predicting sedentary behavior in adults using stacked LSTM modeling, published: 27 November 2024
Volume 16, pages 346–355 (2025), <https://link.springer.com/article/10.1007/s13198-024-02622-2>
53. **Bellary Chiterki Anil, Jayasimha Sondekoppa Rajkumar, T. L. Divya, Samitha khaiyum Rakshitha Kiran P., Balakrishnan Ramadoss**, A Radiomics-based Framework for Liver Cancer Analysis using Explainable Artificial Intelligence (XAI) Methods, Vol. 15 No. 3 (2025): June, 2025, <https://etasr.com/index.php/ETASR/article/view/10377>